第1章

低コストFPGAを用いた高速データ・インターフェースの設計

―1GHzアナログ信号のサンプリングと800MHzのディジタル波形サンプリングを実現する

Minseok Kim

本章では，高速シリアル・データ伝送のLVDS（low voltage differential signaling）インターフェースとクロックの立ち上がり/立ち下がりの両方のエッジを使うDDR（double date rate）について解説する．（編集部）

1. 高速データ・インターフェースを理解しよう

本章ではまず，高速データ・インターフェースの標準として多く使われているLVDS差動インターフェース規格と，リファレンス・クロックの両エッジを用いて転送効率を2倍にすることで，高速データ・レートを実現する方法であるDDRインターフェースについて，市販のFPGAでサポートされる機能を中心に調べてみます．

● 差動データ・インターフェース規格のおさらい

現在，世の中で使われている差動伝送規格にはいろいろありますが，その代表的なものはLVDSという差動インターフェース規格です．これを用いると，プリント基盤線路パターンやケーブルなどのさまざまなメディアを介して，高速データの転送が可能になります．

LVDSは，IEEE standard 1596.3 SCI-LVDSとANSI/TIA/EIA-644の二つの工業標準規格で定義されています．勧告の最大転送速度は，IEEE規格では250Mbps，ANSI/TIA/EIA-644規格では655Mbpsになっています．LVDSインターフェースは，最近のFPGAでは当たり前のように標準インターフェース規格として提供されています．

米国Altera社の場合，ANSI/TIA/EIA-644規格をベースとし，最速6.375Gbpsをサポートするハイエンド製品から，Cyclone IIのように低コストでありながら十分なチャネル数を持つローエンド製品まで製品化しています（**表1**）．また，米国Xilinx社の場合は，Spartan-3という低コスト製品において，344個までの差動インターフェース・ペアを提供しています（**表2**）．

これらの低コストFPGAでは，専用のSerDes（シリアライザ・デシリアライザ；Serializer/Deserializer）回路を内蔵せず，その機能を論理ブロック上に実装することで転送

表1
米国Altera社のFPGAにおける差動インターフェース仕様

ファミリ名	チャネル数	最大データ転送速度	SerDes内蔵
Cyclone	34～129	640 Mbps	なし
Cyclone II	31～261	805 Mbps	なし
Stratix	40～160	840 Mbps	あり
Stratix GX	44～90 （高速チャネル：4～20）	840 Mbps （高速チャネル：3.125 Gbps）	あり
Stratix II	80～312	1.040 Gbps	あり
Stratix II GX	60～144 （高速チャネル：4～20）	1.040 Gbps （高速チャネル：6.375 Gbps）	あり
Stratix III	～264	1.25 Gbps	あり

KeyWord　LVDS，DDR，差動インターフェース，Cyclone II，高速A-Dコンバータ，デマルチプレクサ，AT84AD001B，EP2C8

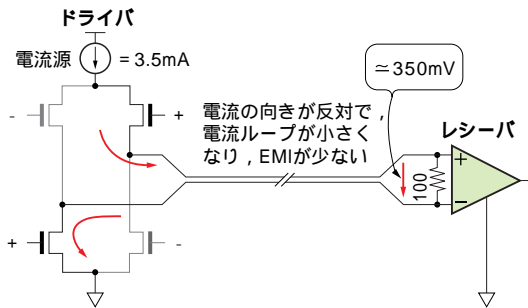



図1 LVDSの動作原理
線路ペアから電流のリターン・パスができるため，電流ループが小さくなり，EMIの影響を少なくできる．

表2 米国Xilinx社のFPGAにおける差動インターフェース仕様

ファミリ名	チャネル数	最大データ転送速度	SerDes内蔵
Spartan 3A	52～227	622 Mbps	なし
Spartan 3E	40～156	622 Mbps	なし
Spartan 3/3L	56～344	622 Mbps	なし
Virtex-4	160～480 （高速チャネル：0～24）	1 Gbps （高速チャネル：10 Gbps）	あり
Virtex-5	200～600 （高速チャネル：0～24）	1.25 Gbps （高速チャネル：3.2 Gbps）	あり

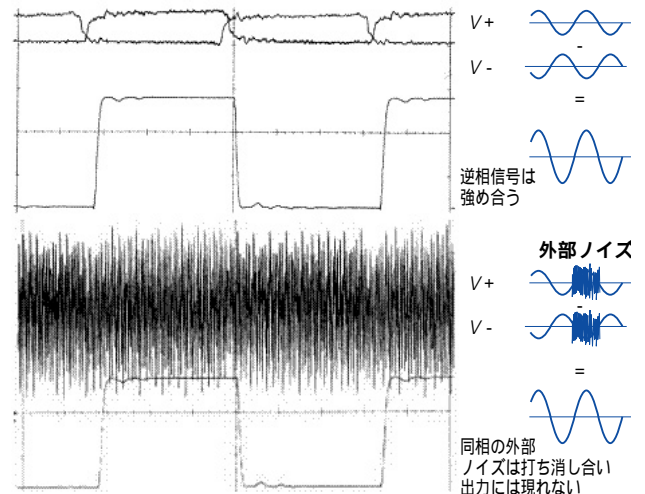



図2[2] 差動信号ペアでは同相ノイズが打ち消し合う（波形データは参考文献（2）を参照）
差動動作により外部からの同相ノイズがうまく打ち消しあい，その出力には差動信号のみが強め合う．

速度は低下しますが，その代わりに低コスト化を実現しています．

Cyclone IIの場合，LVDSから派生したRSDS（reduce swing differential signaling）とmini-LVDSという電圧振幅を小さくし，消費電力を低減した規格についてもサポートしています（出力機能のみ）．さらに，クロック・インターフェースとして多く使われるLVPECL（low voltage positive emitter coupled logic）規格は，FPGAのクロック入力専用ピンでサポートされます．

（1）LVDSの物理的な特性

LVDS規格とシリアライザ・デシリアライザを用いて高速シリアル転送を行うことは，LSI間の配線数が低減でき，基板のコスト削減に直接繋がるという利点があります．さらに，LVDS規格が差動電流モードのインターフェースであることで，外部からのノイズに対して高い耐性を持つと同時に，クロストーク・ノイズも非常に低いものとなります．差動インターフェースは高速データ転送をするときに高い信頼性があることから，数百MHz～数GHzの高速データ転送規格として広範囲に採用されています．

LVDSの物理的な等価回路を**図1**に示します．ドライバ側では，およそ3.5mAの電流をスイッチング動作により，その向きを変化させながら約100Ωの終端抵抗に流します．

このような差動ドライバは，大きさが同じで，符号が反対のオド・モード（odd mode）転送を行います．線路ペアから電流のリターン・パスができるため，電流ループが小さくなります．これはEMI（電磁妨害；electromagnetic interference）の影響を少なくすることができます．

レシーバでは，**図2**のように差動動作によって外部からの同相ノイズ（common-mode noise）がうまく打ち消しあい，その出力には差動信号のみが強め合うことになります．LVDSのノイズ除去効果を最大にするには，線路を同じ長さで対称に配線しなければなりません．また，LVDS線路ペアは，プリント基盤上で可能な限り近くに通るように配線するなど，線路の配線がきわめて重要です．そのほかにも，LVDS線路ペアの配線距離（edge-to-edge）を基準とし，ほかのペアとの距離はその2倍以上にすることや，シングル・エンド線路との距離はその3倍以上離すことなど，LVDS配線におけるいくつかのルールが勧告されています．その詳細については他の文献を参考してください[2]．

近年，データ・レートの増大に伴い，厳しいEMC（電磁的両立性；electromagnetic compatibility）条件から電磁波放射問題がシステム設計において重要な考慮事項になっています．システム設計者は一般に，遠方での電磁波放射

（a） 差動信号線路
電界：お互いに結合され，電気力線が閉じられるので，外に放射されにくくなる
磁界：線路の外側における磁界はキャンセルされる
電磁波の放射が少ない

（b） シングルエンド信号線路
電界：電界が線路から放射される
磁界：電界と垂直に放射される

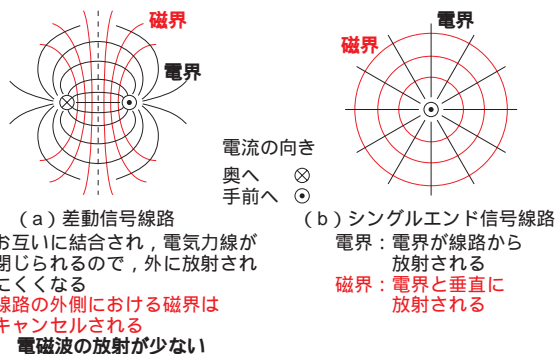



図3　LVDS で電磁波妨害（EMI）が少ない理由

差動信号では，電界は線路ペアの間にうまく結合され，電気力線がお互いに繋がる状態になる．そのとき，磁力線が線路の内側に集中するため，外側に伝搬する電磁界の放射が少なくなる．シングル・エンド信号では，電界は放射線状に発散し，そのとき磁界は円状で線路周りに生成される．これがEMIトラブルの原因となる．

を考慮することになります．電磁波はTEM（transverse electromagnetic）波注1としてシールドを通りぬき，空間に伝搬されることで，さまざまな電磁波妨害を起こすことになります．前述のように，LVDSではこの問題がかなり解決できます．電磁界問題はソース周辺環境との複雑な相好作用を生じるため，予測が難しいとされますが，LVDSの線路ペア構造によって，電磁界を外部に放射しないようにうまく分布させることができます．

図3に，LVDS線路ペアとシングル・エンド線路における電流により生じる電界と磁界を示します．LVDS差動信号の場合，電界は線路ペアの間にうまく結合され，電気力線がお互いに繋がる状態になります．そのとき，線路の外側の磁界はビオサバールの法則注2により打ち消され，磁力線が線路の内側に集中します．このようなことから，外側に伝搬する電磁界の放射が少なくなることが分かります．

注1：電界平面と磁界平面がお互い垂直になり，その進行方向にも垂直な電磁波．
注2：微小電流素の定常電流により生じる磁界を定義する法則．磁界を電流の右回りに周回積分として定義したのがアンペールの法則．
注3：静電荷により生じる力を定義する法則．

それに比べて，CMOS/TTLなどのシングル・エンド線路の場合，電界はクーロンの法則注3により放射線状になり，そのとき磁界は円状で線路周りに生成されます．この電磁界は空間に広がって外部に伝搬して行くことが分かります．これがEMIのトラブルの原因となるわけです．

以上のように，同相ノイズに対する耐性とEMIの低減は，LVDSの物理的な構造から得られる効果です(2)．

(2) FPGAを用いたLVDSのアプリケーション

LVDSはさまざまなアプリケーションに採用されています．ここでは，FPGAを用いた高速データ・コンバータのインターフェースを中心に調べてみます．

最近，計測や無線通信応用などで使われている高速A-Dコンバータ（ADC）やD-Aコンバータ（DAC）は，数GHzのサンプリング・レートをサポートしています．このような高速A-Dコンバータの場合，高速データ・バスとして通常は差動インターフェースを採用し，デマルチプレクサ（demultiplexer；demuxと表記されることが多い）を用いてインタリーブし，相手のLSIの処理可能な速度まで転送レートを下げる仕組みを取ります．

最近のFPGAでは，LVDSレシーバ（あるいはトランスミッタ）を内蔵します．また，ハード・マクロまたはソフト・マクロで提供されるSerDes機能を用いて，データをシリアル-パラレル変換することにより，さらに処理レートを下げることができます．

システム設計者は，SerDesのシリアル-パラレル変換機能およびFPGAが提供するソフトウェア・プログラマビリティを活用すれば，個々の用途に特化した高性能な処理機能を迅速に実現できるわけです．計測や無線通信などの高速データ・インターフェースのアプリケーションでは，FPGAの差動インターフェース機能は，従来使用していた特定用途向けIC（ASIC：application specific integrated circuit）に代わり，広範囲の高性能なアプリケーション分

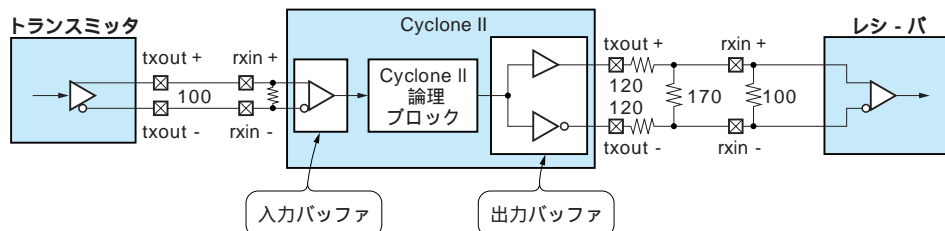
図4
Cyclone IIで差動I/Oの構成
終端抵抗が内蔵されていないため，外部に100Ωで終端する（できるだけピン近くに配置）．

野に手軽に適用することが可能になっています.

図4にCyclone IIの差動インターフェースの構成を示します. LVDSの入力段は100 Ωの抵抗で終端させ, 出力段は抵抗ネットワークで構成します. 抵抗ネットワークはドライバ出力電圧を減衰させ, LVDS規格に合わせるために実装します. FPGAは差動インターフェースバッファだけでなく, 差動終端抵抗を内蔵しているものもありますが, 低コストのCycloneシリーズには内蔵されていません.

● DDRインターフェースのおさらい

LVDS差動インターフェースはデータを高速に転送するための物理的な電気規格であり, SerDesはデータの取り扱いにおける方法論であると言えます. それでは, データ・レートを向上させるもう一つの方法として, リファレンス・クロックの立ち上がりと立ち下がりの両エッジを用いるダブル・データ・レート・インターフェースについて, 低コストFPGAでのサポートを中心に調べてみます.

(1) FPGAのDDRサポートと応用

米国Altera社のFPGAには, altddioというDDRインターフェースのマクロ(以下「DDIOマクロ」と呼ぶ)が提供されています(3). これを用いるとDDR SDRAMのインターフェース実装などを効率的に実現できます. DDIOマクロには, 以下の三つの機能があります.

① altddio_in : リファレンス・クロックの両エッジでデータを受信

② altddio_out : リファレンス・クロックの両エッジでデータを送信

③ altddio_bidir : リファレンス・クロックの両エッジでデータを送受信

このように, DDIOマクロを用いれば, DDRレジスタを用途に合わせて手軽に実装できます. DDRレジスタは, StratixやAPEX IIの場合はI/Oブロック(IOE : IO element)で実装されますが, Cycloneシリーズの場合はピンからいちばん近いロジック・アレイ・ブロック(LAB : Logic Array Block)のロジック・エレメント (LE : Logic element)に実装されます.

DDRレジスタは, データのキャプチャと出力をリファレンス・クロックの両エッジで行います. DDRレジスタはDDR SDRAM, DDR2 SDRAM, RLDRAM II, QDR SRAM, QDRII SRAMなどのメモリ・デバイスとのインターフェースや, LVDSのSerDesにも利用されます. その応用について以下のように整理しました.

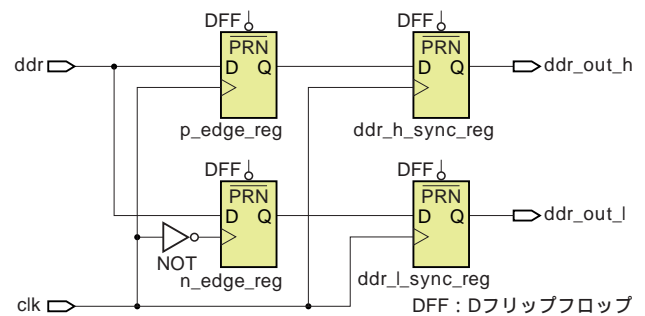



図5 Cycloneシリーズでのダブル・データ・レート入力部の実装
クロックの両エッジで取り込んだデータを, 立ち上がりエッジで読み込む. 立ち下がりエッジで取り込んだデータを先にする場合は, ddr_h_sync_regをバイパスする.

① DDR SDRAM, DDR2 SDRAM, RLDRAM IIのメモリ・インターフェース
クロックの両エッジでデータを取り込むことで, リード・ライト・レートは2倍になります. DDRメモリ・インターフェースはSSTL-II, 1.8-V HSTL, LVTTL, LVCMOSなどのインターフェース規格をサポートします.

② QDR SRAM, QDRII SRAMメモリ・インターフェース
DDRのリード・ポートとライト・ポートを別々に用意し, 通常のSDRAMより4倍の操作が同時に行える仕様になります.

③ 高速のデータ・インターフェース
LVDS, LVPECL, PCML, HyperTransportなどの差動規格において, 高速インターフェースを提供します. このようなデータ転送にもDDRが用いられる場合があります.

④ その他
カスタム・インターフェースに使用します.

(2) DDRレジスタの仕組み

Cycloneシリーズにおいて, DDR入力部は図5のように実装されます.

DDR入力部は, DDR入力ピンからいちばん近いロジック・アレイ・ブロック(LAB)にある四つのロジック・エレメント(LE)のレジスタから構成されます. DDRデータは前段の二つのレジスタ(p_edge_reg, n_edge_reg)にフィードされ, 上部のレジスタ((p_edge_reg)は立ち上が

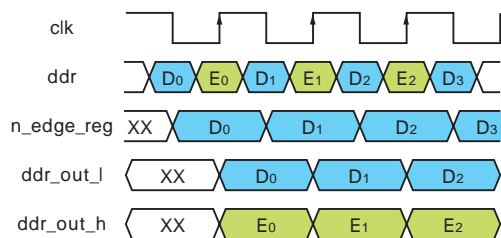



図6　DDR入力部のタイミング・チャート

クロックの両エッジでキャプチャしたデータは，立ち上がりエッジに同期して出力される．

図7　Cycloneシリーズでのダブル・データ・レート出力部の実装

二つのレジスタは二つのシリアル・データ・ストリームを同期させ，データ・レートの2倍のクロックでマルチプレクスする．

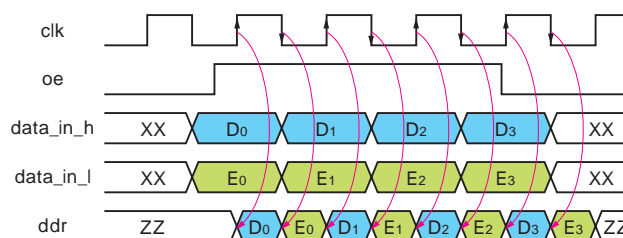



図8　DDR出力部のタイミング・チャート

クロックが“H”(“L”)ならマルチプレクサはdata_in_h（data_in_l）を出力する．

りエッジでキャプチャし，下部のレジスタ（n_edge_reg）は立ち下がりエッジでキャプチャを行います．ddr_h_sync_regとddr_l_sync_regのレジスタは，二つのデータ・ストリームを立ち上がりエッジで同期させます．出力データの順番が，立ち下がりエッジでキャプチャしたデータを先にする場合は，ddr_h_sync_regをバイパスします．

DDR入力部のタイミングを**図6**に示します．これは，StratixやAPEX IIのようにIOEで半クロック・ラッチ（Latch）を用いる方法とは異なります．

図7にCycloneシリーズにおけるDDR出力部を示します．DDR出力部においても，出力ピンからいちばん近いLABのLEが用いられます．二つのレジスタは二つのシリアル・データ・ストリームを同期させ，データ・レートの2倍のクロックでマルチプレクスすることで実現されます．そのタイミングを**図8**に示します．

ここまで，LVDS差動I/O規格とDDRについて，その基本原理とFPGAでのサポートについて述べました．次に，具体的に低コストのFPGA（Cyclone IIファミリのEP2C8F256）を用いて，高速データ・インターフェースを設計してみましょう．まず，LVDSを活用する例として，“1GHz”高速A-Dコンバータのインターフェースのリファレンス設計を紹介し，それからFPGAのDDR機能を用いて“800MHz”の非同期データ・キャプチャを実現する設計方法について解説します．

2．FPGAを用いた高速A-Dコンバータの差動インターフェースの設計

高速A-Dコンバータを用いて回路を設計するには，先に説明したFPGAのLVDS差動インターフェース機能を活用すれば，実現できることが分かりました．次に，そのインターフェース回路を設計してみましょう．

A-Dコンバータは，2チャネル，1Gサンプル/s対応のもの（米国Atmel社のAT84AD001B）を用います．FPGAは低コスト品でありながら，そこそこ高性能であるAltera社のCyclone II（EP2C8F256）を選定しました．ここでは，高い信頼性で高速にデータを取り込んでメモリに格納する機能を，FPGAに実装することになります．

● データ・キャプチャの仕組み

A-Dコンバータの各出力データ・ビットはシリアル信号化され，個別のLVDS差動ペア信号として送信されます．

デシリアライズ動作を容易にするため，シリアル・データ・クロックも同時に出力します．A-Dコンバータから1Gサンプル/s（f_s）でサンプリングされたデータを，**図9**，**図10**に示すように1：2のDEMUXモードで500MHzにレート・ダウンして，LVDSバスからFPGAに入力します．

FPGAのLVDSレシーバ仕様を**表3**に示します．FPGAはスピード・グレード8の場合でも，最大転送レート640Mbpsは可能なので，500MHzは十分に対応可能な帯域であることが分かります．しかし，500MHzの入力データを直接メモリに取り込むことは，FPGA内部のメモリ性能

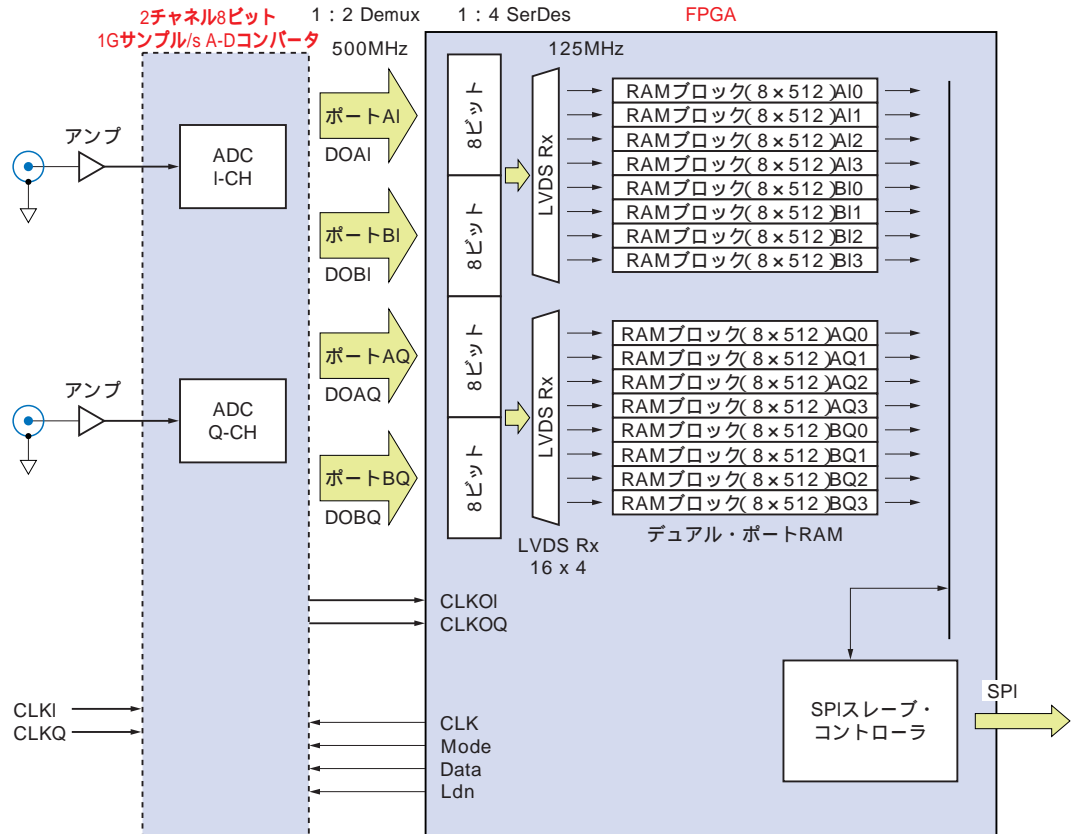



図9
A-Dコンバータの高速データ・インターフェース

最大1Gサンプル/sでサンプリングされたデータは，1：2のDEMUXモードで500MHzにレート・ダウンされ，LVDSバスでFPGAに転送される．

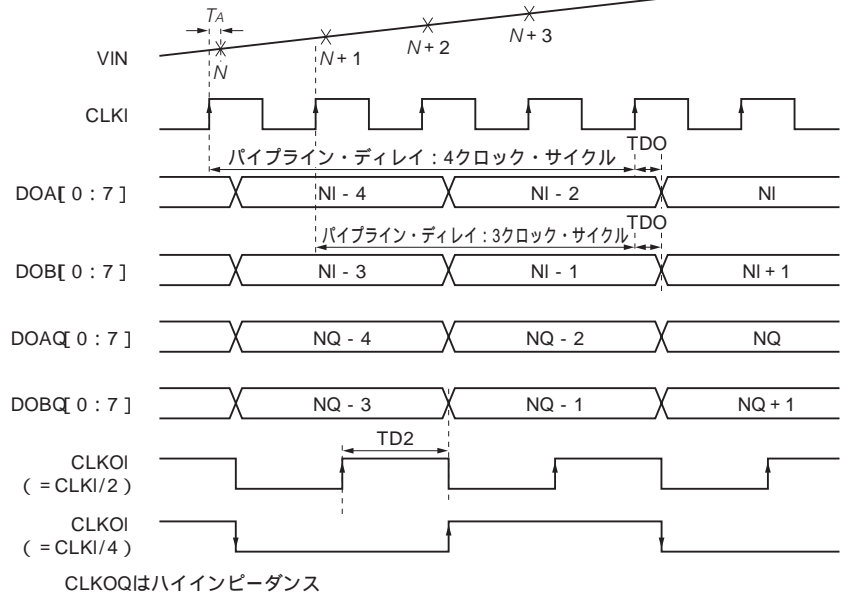



図10
A-Dコンバータ「AT84AD001B」(米国Atmel社)のLVDSデータ・バスのタイミング(1:2 DEMUXモード)

サンプリング・クロックはI/Qチャネル共通(CLKI)にした1：2DEMUXモード．同期クロック出力周波数はFPGA仕様によりサンプリング・クロックの1/4として使用(CLKI/4)．

表3 Cyclone II LVDS レシーバの性能 (1)

記　号	説　明	条 件	－6 スピード・グレード		－7 スピード・グレード		－8 スピード・グレード		単 位
			最小値	最大値	最小値	最大値	最小値	最大値	
f_HSCLK	レシーバの入力クロック周波数	×8	10	402.5	10	320	10	320	MHz
		×4	10	402.5	10	320	10	320	MHz
		×2	10	402.5	10	320	10	320	MHz
		×1	10	402.5	10	402.5	10	402.5	MHz
HSIODR	レシーバの入力データ・レート	×8	80	805	80	640	80	640	MHz
		×4	40	805	40	640	40	640	MHz
		×2	20	805	20	640	20	640	MHz
		×1	10	402.5	10	402.5	10	402.5	MHz
SW	サンプリング・ウィンドウ	―	―	300	―	400	―	400	ps
Input jitter tolerance	PLLがロックするのに要求される入力クロック・ジッタ	―	―	500	―	500	―	550	ps
t_lock	PLLのロック時間	―	―	100	―	100	―	100	ps

を考えると厳しいように思われます．

表4から分かるように，デュアル・ポートRAMをFPGAで実装するには，スピード・グレード7以下の場合に200MHz以下という性能しかありません．従って，ここではAltera社のLVDSマクロのSerDes機能を用いてさらにレート・ダウンを行います．LVDSのSerDesで1：4の×4モードでシリアル-パラレル変換を行い，処理レートを1/4に下げることにします．

これで，FPGAの内部メモリが十分に動作可能なレートである125MHz（500MHz÷4）まで，デュアル・ポートRAMの書き込みクロック・レートを下げることが可能になります．つまり，A-Dコンバータのマルチプレクス動作とLVDSレシーバのSerDesを用いることから，サンプリング周波数の1/8のレートでメモリ書き込みが行われます．

コラム　GHz-サンプリングA-Dコンバータの選択

最近，テスト・ソリューションや計測機器，無線通信業界におけるデータ変換は，GHzの時代に入りつつあります．これほど高速にデータ処理のできるシステムを設計することは，アナログ感覚を持ったハードウェア設計，FPGAのインターフェース，メモリ，ロジック設計，信号処理，アプリケーション・ソフトウェアの開発にいたるまで、さまざまなエンジニアリング・スキルが必要になることを意味します．

表Aに現在市販されているGHzサンプリングのA-Dコンバータをいくつか挙げてみました．これらには，高速の差動データ・バス・インターフェース規格であるLVDSやECL（emitter coupling logic）が採用されていることが分かります．ECLやPECL（positive referenced ECL）で実現されているA-Dコンバータは宿命的に消費電力

表A　市販されているGHz-A-Dコンバータ（2006年　現在）

型　名	メーカ	データ規格	分解能（ビット）	チャネル数	サンプルレート（GHz）	消費電力（W）	帯域（GHz）	電源電圧（V）
MAX104	Maxim	LVPECL	8	1	1	5.25	2.2	5，3.3，－5
MAX108	Maxim	LVPECL	8	1	1.5	5.25	2.2	5，3.3，－5
ADC08D1500	NS	LVDS	8	2	1.5	1.9	1.7	1.9
ADC08D1000	NS	LVDS	8	2	1	1.6	1.7	1.9
AT84AD001B	Atmel	LVDS	8	2	1	1.4	1.5	3.3，2.25
AT84AS003	Atmel	LVDS	10	1	1.5	6.5	3	3.3，2.5，－5，－2.2
AT84AS004	Atmel	LVDS	10	1	2	6.5	3	3.3，2.5，－5，－2.2
AT84AS008	Atmel	ELC/LVDS	10	1	2.2	4.2	3.3	3.3，1.5，5，－5，－2.2
TS83102G0B	Atmel	ELC/LVDS	10	1	2	4.6	3.3	5，－5，－0.8/1.45
TS83102G0BMGS	Atmel	ELC/LVDS	10	1	2	4.6	3.3	5，－5，－0.8/1.45

Maxim：Maxim Integrated Products社　NS：National Semiconductor社　Atmel：Atmel Coporation社

表4 Cyclone IIの内部メモリの動作スピード(1)

応用例		使用するリソース			パフォーマンス			
		LE数	M4Kメモリ・ブロック	DSPブロック	−6スピード・グレード	−7スピード・グレード	−8スピード・グレード	単位
メモリM4Kブロック	単純なデュアル・ポートRAM 128×36ビット	0	1	0	235.29	194.93	163.13	MHz
	真のデュアル・ポートRAM 128×18ビット	0	1	0	235.29	194.93	163.13	MHz
	FIFO 128×16ビット	32	1	0	235.29	194.93	163.13	MHz
	単純なデュアル・ポートRAM 128×36ビット	0	1	0	210.08	195.0	163.02	MHz
	真のデュアル・ポートRAM 128×18ビット	0	1	0	163.02	163.02	163.02	MHz

そのかわりに，8個メモリを用いて，同時に書き込みを行うことになります．

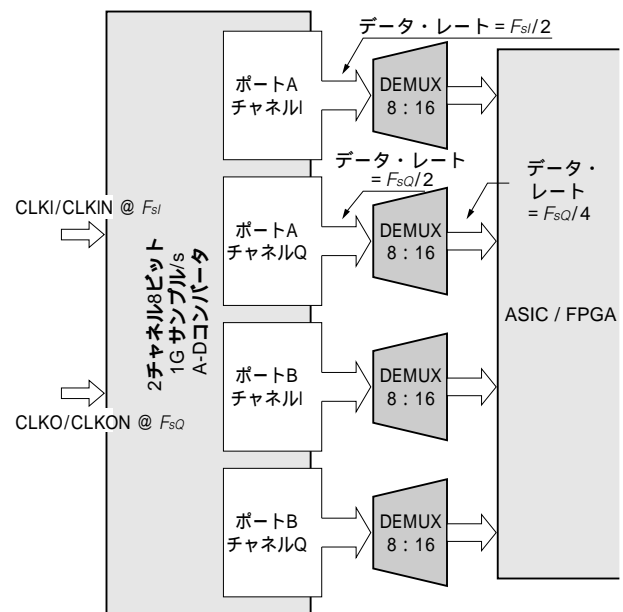
LVDSレシーバのSerDes出力は，八つの異なるタイミングのデータが64ビット（8ビット×8）のワード形式として出力されます．これを**図11**のようにビットの順番を揃えてバッファリングを行い，それからLVDSレシーバの同期出力クロック（outclock）に同期して各メモリに書き込みます．今回用いるA-Dコンバータのシリアル・データ・クロックの仕様は，サンプリング・クロックの1/2と1/4である500MHzと250MHzとして設定するようになっています．しかし，**表3**のようにLVDSレシーバ仕様上の受信同期クロック（inclock）は×4モードで402.5MHzまでに制限されるため，シリアル・データ・クロックを250MHz（1/4レート）に設定しました．これは，FPGAの内部PLL（phase-locked loop）の1個（2個内蔵）をLVDS専用として使用し，そのクロック管理機能から，SerDesに必要なクロックを生成することになります．

● LVDS機能を用いた論理回路の設計

前述のように，低コストFPGAには専用のSerDes回路が内蔵されていません．この場合，高速インターフェースを実現するには，高性能のSerDes回路の実装が必須です．

が高く，その応用が制限されてしまいます．また，負電源が必要になるなど，システム設計者にとっては非常に扱いにくいものでした．

しかし，米国National Semiconductor社や米国Atmel社から発売されたものは、8ビット精度で1GHz以上の性能があります．特にAtmel社2チャネル品であるAT84AD001Bの場合，BiCMOS技術で製造され1.4W程度（1チャネル当たり0.7W）の低消費電力化を達成しています．一般に，GHzサンプリングA-Dコンバータの出力データを直接相手のLSIに転送することは無理であり，デマルチプレクサによりレート・ダウンすることになります．

図AにAtmel社のA-Dコンバータの，LVDSデータ・バス・インターフェースの動作例を示します．1Gサンプル/sでA-D変換されたデータは，内蔵のデマルチプレクサから，ポートAとポートBを用いて半分のデータ・レートで出力されます．また，外部のデマルチプレクサICを使用してさらに半分のデータ・レートに落とし，相手のLSIには出力データが1/4のリーズナブルなデータ・レート（250Mサンプル/s）で渡されることになります．高速のデータを扱う場合，FPGAの差動インターフェースサポートやLVDSのトランシーバ機能は，このようなインターフェースを設計する際に非常に役に立ちます．

図A 米国Atmel社のA-Dコンバータ「AT84AD001B」のデータ・インターフェース

1Gサンプル/sでA-D変換されたデータは，内蔵のデマルチプレクサからポートAとポートBを用いて半分のデータ・レートで出力される．さらに外部のデマルチプレクサIC（DEMUX）を使用し，相手のLSIには出力データが1/4のデータ・レート（250MHz）で渡される．

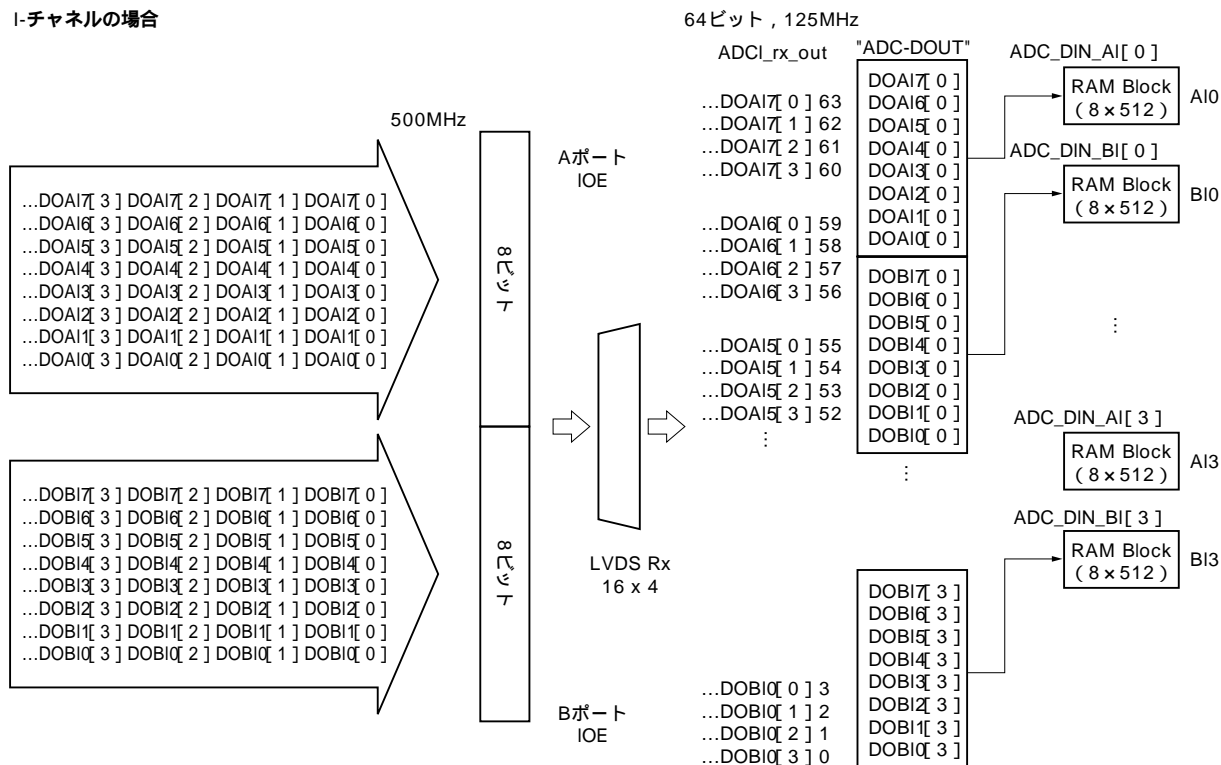



図11　LVDSのSerDesを用いたシリアル-パラレル変換とメモリ・インターフェース
A-Dコンバータのマルチプレクス動作とLVDSレシーバのSerDesより，サンプリング周波数の1/8のレートで8個のメモリを用いて同時に書き込みが行われる．八つの異なるタイミングのデータが64ビット（8ビット×8）のワード形式として出力され，そのビットの順番を揃えて各メモリに書き込まれる．

これは，設計者が論理ブロック上に実装することになります．しかし，Altera社の設計ツールのQuartus IIで提供されているALTLVDSというマクロを使えば，高速のSerDesを簡単に実現することができます．

図12と**図13**に，Quartus II バージョン6.0（Web Edition, 無償版）のMegaWizard Plug-In ManagerのALTLVDSを実行した画面を示します．ここで，今回の仕様にあわせてレシーバの設計を行ってみます．

図12の設定項目で，チャネル数（number of channels）はA-Dコンバータで1：2にデマルチプレクスされた「16」

コラム　FPGAの選定

本特集は，FPGAを用いて高速A-DコンバータのLVDS差動インターフェースとDDRデータ・キャプチャの活用法をテーマとしています．理解を助けるためにターゲットFPGAを選定し，具体的な設計事例を紹介します．今回は，可能な限り低価格のもので実現することを目標にし，米国Altera社のCyclone IIを選定することにしました．また，選定上，以下のようなポイントを考慮しました．

①LVDS差動チャネル数を含めたポート数
A-Dコンバータのインターフェース用のLVDSチャネル（データ32本＋クロック2本＋リセットなど＝35～37本程度）とDDRディジタル・キャプチャ用の入力ポート（16ビット）．

②LVDSレシーバの転送性能
500Mbps以上．

③PLL数
最低2個（LVDSとFPGA内部のクロック・マネジメントに使用）．

④ゲート容量
実装に十分なLE数．

⑤内部メモリ容量
多いほどよいのですが，とりあえずチャネルごとに4,096点程度．

以上の条件で，Altera社Cyclone II（EP2C8F256，スピード・グレード8）は，77本の差動チャネル，最大640Mbps程度のLVDSレシーバ転送性能，二つのPLL，8,256 LE，165,888ビットの内部メモリという仕様なので，想定した機能を実現するには最小限のものでしょう．

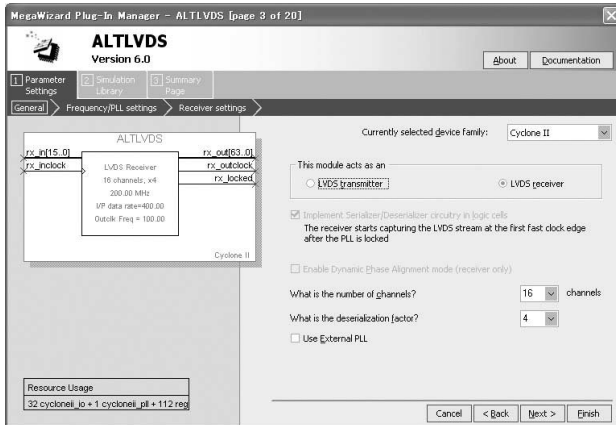



図12 ALTLVDS MegaWizard (1)

Quartus II Web EditionのMegaWizard Plug-In Managerを用いてLVDSレシーバの設計を行う．チャネル数（Number on channels）はA-Dコンバータで1：2にデマルチプレクスされた“16”チャネルになり，デシリアライズ・ファクタ（Desirialization factor）は，それぞれのチャネルに対して1：4にレート・ダウンするため“4”になる．

図13 ALTLVDS MegaWizard (2)

A-Dコンバータからのデータ・レートは1：2でデマルチプレクスされて“500”Mbpsになる．入力クロック・レートはA-Dコンバータの仕様に合わせて250MHz (f_s/4) とする．

チャネル（ビット）になり，デシリアライズ・ファクタ（desirialization factor）は，それぞれのチャネルに対してSerDesで1：4にレート・ダウンするので，「4」になります．**図13**では転送レートの設定を行います．

A-Dコンバータからのデータ・レートは1：2でデマルチプレクスされて500Mbpsになるので，入力データ・レートは500Mbpsとなります．Cyclone IIでは入力クロック・レートが×4モードで最大250MHzまで設定可能で，A-Dコンバータの仕様に合わせて250MHz (f_s/4) とします．クロックとデータのタイミングについても，位相調整の設定項目（phase alignment）からPLLを用いて，仕様に合わせた最適なデータ・キャプチャを行うことができるようになっていますが，今回は特に設定を行いません．

今回，LVDSを用いて設計したA-Dコンバータのインターフェース回路（lvdsserdes.vhd）を，VHDLで記述したものを**リスト1**に示します．**図11**のように，16ビットのデマルチプレクスされた入力データをLVDSのSerDesから64ビットに展開して，八つのメモリに1/8レート（125MHz）で並列に書き込む動作を記述しています．

メモリの実装はLVDSと同様に，Altera社のRAM:2-Portマクロを使用しました．ここで，FIFOで実装してもかまいませんが，今回はアドレス操作が可能なデュアル・ポートRAMとしました．

なお，Cyclone IIには終端抵抗が内蔵されていないため，ピン近くに100Ωの抵抗で終端する必要があります．LVDSポートの設定は，ポジティブ・ピンとネガティブ・ピンの両方をLVDSとして割り当てる必要があります．

	From	To	Assignment Name	Value	Enabled
16		clkp[0]	I/O Standard	LVTTL	Yes
17		clkp[2]	I/O Standard	LVDS	Yes
18		clkp[2](n)	I/O Standard	LVDS	Yes
19		DDRB	I/O Standard	LVDS	Yes
20		DDRB(n)	I/O Standard	LVDS	Yes
21		DOAI[0]	I/O Standard	LVDS	Yes
22		DOAI[0](n)	I/O Standard	LVDS	Yes
23		DOAI[1]	I/O Standard	LVDS	Yes
24		DOAI[1](n)	I/O Standard	LVDS	Yes

図14 Quartus II 6.0のAssignment Manager上のI/O規格の設定

LVDSポートの設定はポジティブ・ピンやネガティブ・ピン両方をLVDSとしてアサインする．

図14にQuartus IIのAssignment Manager上，そのインターフェース規格を設定した画面を示します．

3．FPGAを用いたDDRデータ・キャプチャの設計

それでは，FPGAを活用してダブル・データ・レート・キャプチャを設計してみましょう．

今回，設計例としたのは，FPGAの入力ポートの論理値を高速でメモリに取り込む機能（FPGA内部クロックを用いる非同期キャプチャ）になります．具体的にこれを用いると，FPGAで「ロジック・アナライザ機能」を実現することが可能です．今回の設計目標は選定した低コストFPGA上で，実現可能な最高の性能を目指します．選定したFPGAを用いることを想定すると，その性能は以下のようになります．

① 内部PLLから論理ブロックに供給可能なクロック周波

リスト1
LVDSを用いたA-Dコンバータのインターフェース記述(lvdsserdes.vhd)

16ビットのDEMUXされた入力データをLVDSのSerDesから64ビットに展開して，八つのメモリに1/8レート(125MHz)で並列に書き込むためのデータを用意する．

```
--************************************--
--       LVDSSERDES.vhd
--        FIFO 1:4 Mode
--    (ADC DEMUX ratio is 1:2)
--       by Minseok Kim
--************************************--

LIBRARY IEEE;
USE  IEEE.std_logic_1164.all;
PACKAGE type_package IS
  TYPE slv8ar4  IS ARRAY(1 to 4)  OF std_logic_vector(7 downto 0);
END type_package;

LIBRARY IEEE;
USE IEEE.std_logic_1164.all;
USE IEEE.std_logic_arith.ALL;
USE IEEE.std_logic_UNSIGNED.all;
USE work.util_package.ALL;

ENTITY LVDSSERDES IS
  PORT (
    inclock       :IN std_logic;
    ADC_rx_data   :IN std_logic_vector(15 downto 0);

    outclock      :OUT std_logic;
    locked        :OUT std_logic;
    ADC_DOUT_A    :OUT slv8ar4;
    ADC_DOUT_B    :OUT slv8ar4
);
END ENTITY LVDSSERDES;

ARCHITECTURE rtl OF LVDSSERDES IS

  COMPONENT ALTLVDS
     PORT
     (
              rx_in              : IN STD_LOGIC_VECTOR (15 DOWNTO 0);
              rx_inclock         : IN STD_LOGIC  := '0';
              rx_out             : OUT STD_LOGIC_VECTOR (63 DOWNTO 0);
              rx_outclock        : OUT STD_LOGIC;
              rx_locked : OUT STD_LOGIC
     );
  END COMPONENT;

  SIGNAL  ADC_rx_out               : std_logic_vector(63 downto 0);

BEGIN

-- CLKI  __|~~|__|~~|__|~~|__|~~|__|~~
-- DOIA    >|<  NI-4   >|<  NI-2   >|<
-- DOIB    >|<  NI-3   >|<  NI-1   >|<
-- DOQA    >|<  NQ-4   >|<  NQ-2   >|<
-- DOQB    >|<  NQ-3   >|<  NQ-1   >|<
-- CLKOI  ~~|___________|~~~~~~~~~~~|  (1/4fs)

   -- 図 12，図 13で作られたLVDSレシーバ ALTLVDSマクロ
    altera_lvds_ADC : ALTLVDS PORT MAP
            (rx_in      => ADC_rx_data,
            rx_inclock  => inclock,
            rx_out      => ADC_rx_out,
            rx_outclock => outclock,
            rx_locked   => locked);


--  RAMに取り込むデータワードの作り(Aポート4個+Bポート4個で，8個データを出力)
    outdata_loop: FOR i IN 0 TO 3 GENERATE

    ADC_DOUT_A(i+1) <=
                ADC_rx_out(63-i) & ADC_rx_out(59-i) & ADC_rx_out(55-i) & ADC_rx_out(51-i) &
                ADC_rx_out(47-i) & ADC_rx_out(43-i) & ADC_rx_out(39-i) & ADC_rx_out(35-i);

    ADC_DOUT_B(i+1) <=
                ADC_rx_out(31-i) & ADC_rx_out(27-i) & ADC_rx_out(23-i) & ADC_rx_out(19-i) &
                ADC_rx_out(15-i) & ADC_rx_out(11-i) & ADC_rx_out( 7-i) & ADC_rx_out( 3-i);

    END GENERATE;


END ;
```

数：402.5 MHz以下（**表5**から）

② 内部メモリの動作周波数：約200MHz以下（**表4**から）

③ PLLの数量：2個

以上のことから，ディジタル入力に対してPLLの最大周波数である402.5MHz（グレード8）のダブル・レートでデータのキャプチャが行えるなら，800Mサンプル/s程度の性能を得られそうです．つまり，800Mサンプル/sのキャプチャを達成するには，FPGAの内部PLLからほぼ最大出力周波数である400MHzのクロックを生成し，その立ち上がり/立ち下がりの両エッジでキャプチャを行うことになります．

また，メモリの動作クロック周波数についても，LVDSの場合と同様に，書き込みレートを正常動作可能な周波数まで下げる必要があります．これは，**図15**のように複数の異なる位相（例えば，0度，90度，180度，270度など）のクロック（あるいはライト・イネーブル）を生成し，それぞれの位相に対応するメモリに取り込む動作になります．この場合，PLLを用いて複数のクロックの位相関係保証やFPGAのグローバル・クロック数も限られていることから，低コストFPGAでの設計が難しいと考えられます．

また，応用によってキャプチャ・レートを可変にするために，ベース・クロックのレートを切り替える必要が出てきます．しかし，ベース・クロックが切り替えられたときに，複数のクロックそれぞれの位相関係が崩れないように対応しないといけないなど，その方法ではいろいろな問題がありそうです．

● データ・キャプチャの仕組み

最終的に，どういう仕組みで800Mサンプル/sといった高速キャプチャが実現できるのでしょうか．

図16に，FPGAのディジタル入力ポート（POD）から複数のメモリ・ブロックに書き込みを行うまでの構成を示します．PODの電圧レベルはFPGAのLVTTL I/Oバッファにより3.3Vロジックで“H”が“L”という論理値として現れます．その論理値は，DDRバッファによりクロック（POD

表5 Cyclone IIのPLL仕様(1)

記　号	パラメータ	最小値	最大値	単　位
fin	入力周波数（−6スピード・グレード）	10		MHz
	入力周波数（−7スピード・グレード）	10		MHz
	入力周波数（−8スピード・グレード）	10		MHz
fin Duty	入力クロック・デューティ・サイクル	40	60	MHz
fout_ext（外部PLLクロック出力）	PLL出力周波数（−6スピード・グレード）	10		MHz
	PLL出力周波数（−7スピード・グレード）	10		MHz
	PLL出力周波数（−8スピード・グレード）	10		MHz
fout（グローバル・クロックへ）	PLL出力周波数（−6スピード・グレード）	10	500	MHz
	PLL出力周波数（−7スピード・グレード）	10	450	MHz
	PLL出力周波数（−8スピード・グレード）	10	402.5	MHz
f_vco	PLLinside（内部）VCO動作範囲	300	1000	MHz

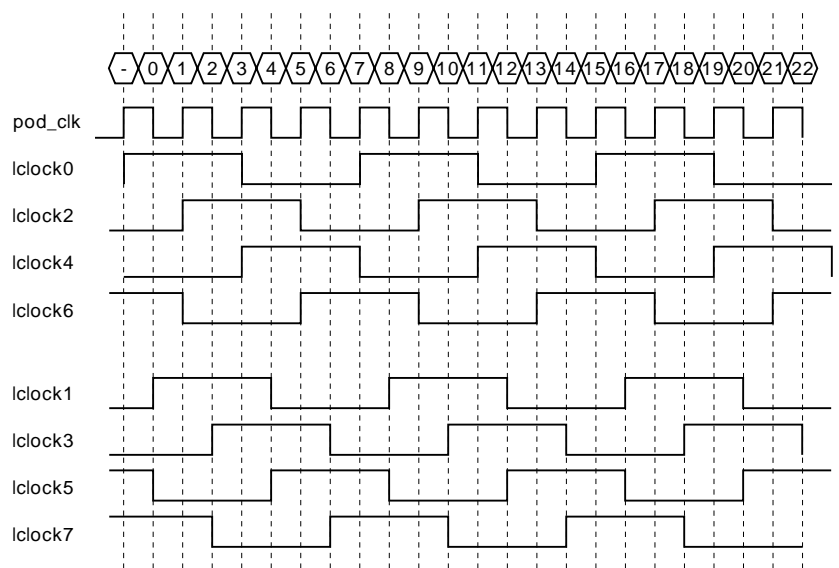



図15 位相の異なる複数のクロックを用いたデータ・キャプチャ
複数の異なる八つの位相（0, 45，90, 135, 180, 225, 270, 315度）のクロックを生成し，それぞれの位相に対応するメモリに取り込む動作．低コストFPGAではクロック位相の管理が難しい．

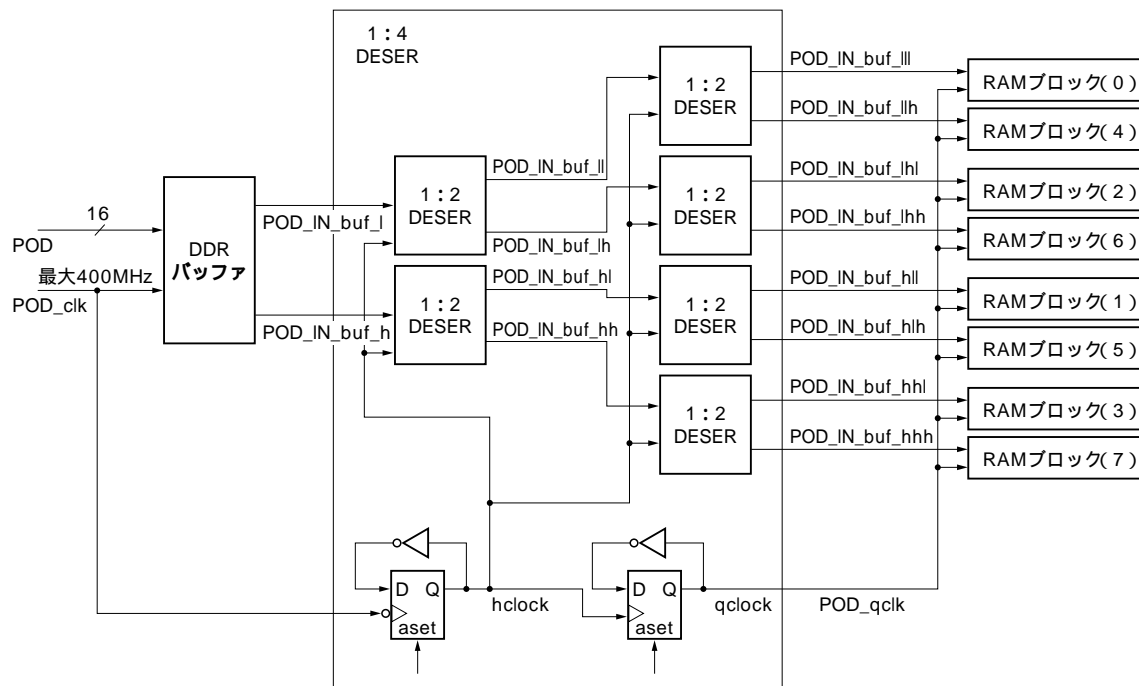



図16　並列メモリ書き込みの仕組み

ALTDDIO_INを用いるとDDRバッファが入力ポート近くの最適な場所に生成される．1：2デシリアライザを2段にすることで2チャネルの1：4デシリアライザを構成し，各データをPOD_qclkに同期して八つのメモリに書き込む．

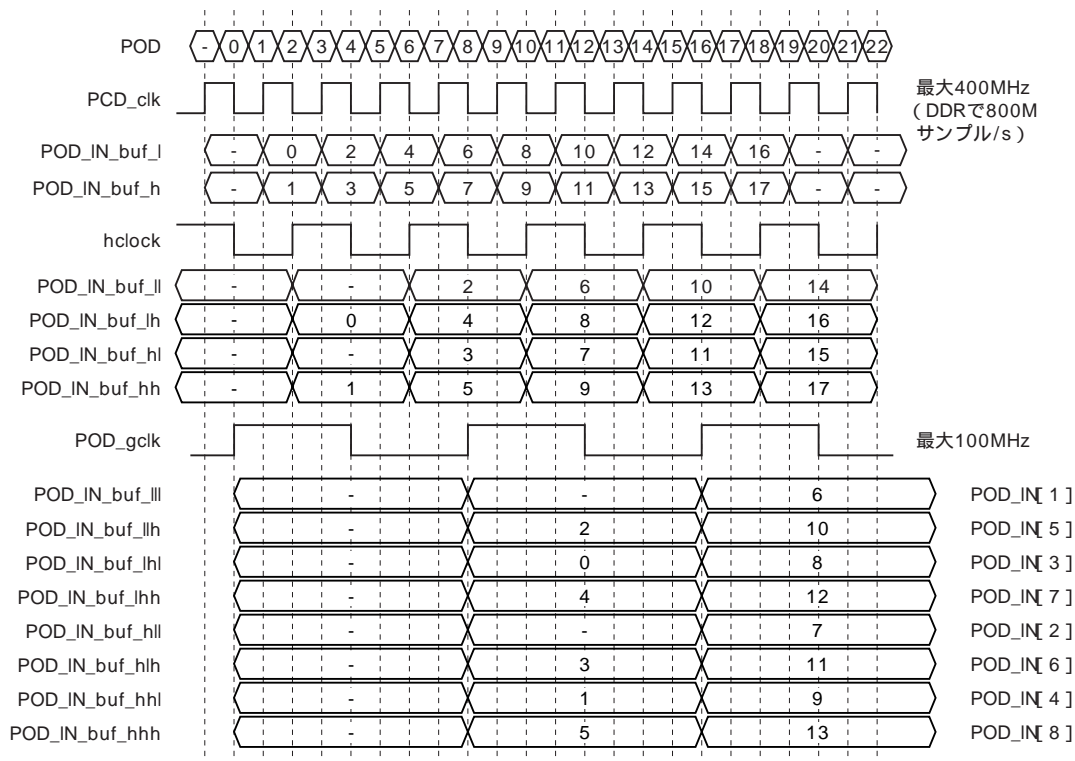



図17
並列メモリ書き込みタイミング

PLLから生成されたPOD_clk（Max400MHz）に同期してDDRバッファに取り込み，それからデシリアライザによりレート・ダウンを行う．

_clk)の立ち上がり/立ち下がりの両エッジでキャプチャされ，それぞれのデータはクロックの立ち上がりエッジで同期出力されます．ここで，POD_clkが最大400MHzであれば，キャプチャ・レートは800Mサンプル/sになります．

DDRバッファはAltera社のDDIOマクロ(ALTDDIO_IN)を用いて実装できます．DDRバッファで両エッジにキャプチャされたデータは1：4のDESER(deserializer)モジュールに入力され，最終的に八つの異なる位相でキャプチャしたデータが得られます．すべての出力データは，POD_qclkの立ち上がりエッジで同期しているため，メモリの実装が簡単になります．

図17に**図16**の構成のタイミングを示します．このように，DDIOマクロを用いたダブル・データ・レート・キャ

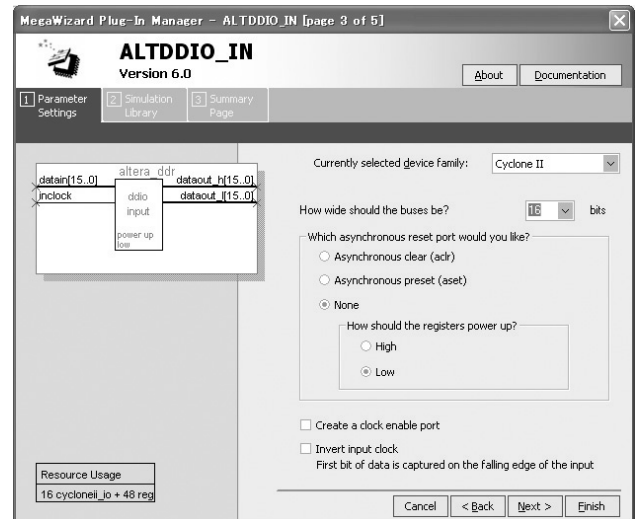



図18 MegaWizardによるALTDDIO_INの設定画面
入力のワード長のみ設定すればOK．

リスト2
16ビット1：2デシリアライザのVHDL記述(DESERx2.vhd)
16ビット・データをクロックの両エッジで取得し，立ち上がりエッジに同期して出力する．

```
--************************************--
--        deserx2.vhd
--      by Minseok Kim
--************************************--

LIBRARY IEEE;
USE IEEE.std_logic_1164.all;
USE work.util_package.ALL;

ENTITY DESERx2 IS
  PORT (
    DIN         :IN std_logic_vector(15 downto 0);
    inclock     :IN std_logic;
    OUTDATAL    :OUT std_logic_vector(15 downto 0);
    OUTDATAH    :OUT std_logic_vector(15 downto 0)
);
END ENTITY DESERx2;

ARCHITECTURE rtl OF DESERx2 IS

  SIGNAL n_edge_reg, p_edge_reg  : std_logic_vector(15 downto 0);
  SIGNAL ddr_l_sync_reg, ddr_h_sync_reg  : std_logic_vector(15 downto 0);

BEGIN

   -- 立ち上がりエッジでキャプチャ
    PROCESS (inclock) IS
      BEGIN
        IF (inclock'event AND inclock ='1') THEN
          p_edge_reg <= DIN;
        END IF;
      END PROCESS;

   -- 立ち下がりエッジでキャプチャ
    PROCESS (inclock) IS
      BEGIN
        IF (inclock'event AND inclock ='0') THEN
          n_edge_reg <= DIN;
        END IF;
      END PROCESS;

   --  立ち下がりエッジでキャプチャしたデータは半クロック遅延
    PROCESS (inclock) IS
      BEGIN
        IF (inclock'event AND inclock ='1') THEN
          ddr_l_sync_reg <= n_edge_reg;
        END IF;
      END PROCESS;

    OUTDATAH(15 downto 0) <= p_edge_reg;
    OUTDATAL(15 downto 0) <= ddr_l_sync_reg;

END ;
```

リスト3　16ビット1：4デシリアライザのVHDL記述(deserx4.vhd)
2入力の16ビット・データを1：4にデシリアライズし，8ワード(128ビット)データを時間順に並べ変えて出力する．

```
--*************************************--
--        deserx4.vhd
--       by Minseok Kim
--*************************************--


LIBRARY IEEE;
USE IEEE.std_logic_1164.all;
USE work.util_package.ALL;

ENTITY DESERx4 IS
  PORT (
    DIN_L       :IN std_logic_vector(15 downto 0);
    DIN_H       :IN std_logic_vector(15 downto 0);
    inclock     :IN std_logic;
    outclk      :OUT std_logic;
    reset       :IN std_logic;
    OUTDATA     :OUT std_logic_vector(127 downto 0)
);
END ENTITY DESERx4;

ARCHITECTURE rtl OF DESERx4 IS

  COMPONENT DESERx2 IS
    PORT (
      DIN         :IN std_logic_vector(15 downto 0);
      inclock     :IN std_logic;
      OUTDATAL    :OUT std_logic_vector(15 downto 0);
      OUTDATAH    :OUT std_logic_vector(15 downto 0)
  );
  END COMPONENT;

  SIGNAL  hclock, qclock, reset_sig                 : std_logic;
  SIGNAL  dout_LL, dout_LH, dout_HL, dout_HH        : std_logic_vector(15 downto 0);
  SIGNAL  dout_LLL, dout_LLH, dout_LHL, dout_LHH    : std_logic_vector(15 downto 0);
  SIGNAL  dout_HLL, dout_HLH, dout_HHL, dout_HHH    : std_logic_vector(15 downto 0);

  BEGIN

   --  非同期セット付きDFFからハーフ・クロックを生成
    PROCESS (inclock) IS
      BEGIN
        IF reset = '1' THEN
          hclock <= '1';
        ELSIF inclock'event and inclock = '0' THEN
          hclock <= not hclock;
        END IF;
      END PROCESS;
```

プチャ機能の活用とDESERモジュールの設計により，八つのデータは100MHz(キャプチャ・レートの1/8)のクロックで，それぞれのメモリに対して書き込まれることが分かります．このようにすれば，低コストのFPGAを用いても最大800Mサンプル/sの高速のデータ・キャプチャが実現できます．

● DDIOマクロを用いた論理回路の設計

FPGAのDDR機能は，Altera社のMegaWizardを用いてALTDDIO_INマクロを実装すれば手軽に実装できます．このマクロの入力はFPGAの入力ピンのみになります．これを用いると，ダブル・データレート・バッファは入力ポートの近くに効率良く配置されます．

図18にMegaWizardの設定画面を示します．普通の場合なら，設定するのは入力のワード長のみになります．

さらに，**図16**のようにデシリアライザ(DESER)を実装する必要があります．DDIOマクロを利用したいところですが，残念ながら入力ポートがピンのみに限定されています．従って，**図5**のような回路を直接論理ブロック上に実装することにしました．

リスト2と**リスト3**は，それぞれ16ビット入力の1：2(deserx2.vhd)と，2入力1：4デシリアライザ(deserx4.vhd)のVHDL記述を示しています．**リスト3**では，2入力の16ビット・データを1：4にデシリアライズし，8ワード(128ビット)データを時間順に並べ変えて出力します．

リスト4では，**図16**のようにメモリ書き込み用のデータ・バッファまでの回路を記述しています．また，高速回路設計の場合，開発ツール(例えば，Quartus IIなど)を用

```
    -- リセット信号をハーフクロック遅延
    PROCESS (hclock) IS
      BEGIN
        IF hclock'event and hclock = '0' THEN
          reset_sig <= reset;
        END IF;
      END PROCESS;

    -- 非同期セット付きDFFからクォータ・クロックを生成
    PROCESS (hclock) IS
      BEGIN
        IF reset_sig = '1' THEN
          qclock <= '0';
        ELSIF hclock'event and hclock = '0' THEN
          qclock <= not qclock;
        END IF;
      END PROCESS;

    outclk <= qclock;

    -- 1st ステージ
    DESERx2_stage_L  : DESERx2 PORT MAP (DIN      => DIN_L, inclock => hclock,
                                         OUTDATAL => dout_LL, OUTDATAH => dout_LH);
    DESERx2_stage_H  : DESERx2 PORT MAP (DIN      => DIN_H, inclock => hclock,
                                         OUTDATAL => dout_HL, OUTDATAH => dout_HH);

    -- 2nd ステージ
    DESERx2_stage_LL : DESERx2 PORT MAP (DIN      => dout_LL, inclock => qclock,
                                         OUTDATAL => dout_LLL, OUTDATAH => dout_LLH);
    DESERx2_stage_LH : DESERx2 PORT MAP (DIN      => dout_LH, inclock => qclock,
                                         OUTDATAL => dout_LHL, OUTDATAH => dout_LHH);
    DESERx2_stage_HL : DESERx2 PORT MAP (DIN      => dout_HL, inclock => qclock,
                                         OUTDATAL => dout_HLL, OUTDATAH => dout_HLH);
    DESERx2_stage_HH : DESERx2 PORT MAP (DIN      => dout_HH, inclock => qclock,
                                         OUTDATAL => dout_HHL, OUTDATAH => dout_HHH);

    OUTDATA <= dout_HHH &  -- 8th
               dout_LHH &  -- 7th
               dout_HLH &  -- 6th
               dout_LLH &  -- 5th
               dout_HHL &  -- 4th
               dout_LHL &  -- 3rd
               dout_HLL &  -- 2nd
               dout_LLL;   -- 1st
END ;
```

いて論理合成と配置配線を行う際に，クロックになる信号には正しく周波数設定をすることは必須なので，要注意です．

Minseok Kim
(株)ブレインズ

参考・引用*文献

(1) Altera；Cyclone II Device Handbook, Volume 1, 2006.
(2) *National Semiconductor；“LVDS Owner's Manual,”3rd Edition (http://lvds.national.com/), 2004.
(3) Altera；“altddio Megafunction User Guide,” Jul. 2006.
(4) 西本桃子；低コストFPGAによる640Mbps LVDSインターフェースの実現，Design Wave Magazine，2005年9月号，pp.108-115.

<筆者プロフィール>
金 旼錫(Minseok Kim)．韓国Hanyang大学 電気工学科卒．横浜国立大学 大学院 博士課程修了．ブレインズ入社以来，組み込みシステム・ベースの電子機器開発に従事．現在，横浜国立大学にて，スマート・アンテナ(Smart Antenna)やソフトウェア無線(Software Defined Radio)に関わる研究に従事．

リスト4
DDIOマクロを用いたデータ・キャプチャ・インターフェース記述（top.vhdから抜粋）

図16のようなメモリ書き込み用データ（POD_IN）を作成するまでの回路．

```
（中略）

-- ALTDDIO_INを用いて構成したDDIOコンポーネント
COMPONENT altera_ddr IS
      PORT
      (
               datain           : IN STD_LOGIC_VECTOR (15 DOWNTO 0);
               inclock          : IN STD_LOGIC ;
               dataout_h        : OUT STD_LOGIC_VECTOR (15 DOWNTO 0);
               dataout_l        : OUT STD_LOGIC_VECTOR (15 DOWNTO 0)
      );
END COMPONENT;

-- 16ビット1：4デシリアライザ・コンポーネント
COMPONENT DESERx4 IS
  PORT (
    DIN_L               :IN std_logic_vector(15 downto 0);
    DIN_H               :IN std_logic_vector(15 downto 0);
    Inclock    :IN std_logic;
    Outclk     :OUT std_logic;
    Reset               :IN std_logic;
    OUTDATA    :OUT std_logic_vector(127 downto 0)
);
END COMPONENT;

（中略）

---- DDIOとDESERx4を用いた1：8デシリアライザ  ============================================

---- POD                  ...>|<D0 >|<D1 >|<D2 >|<D3 >|<D4 >|<D5 >|<D6 >|
---- POD_clk  __|~~~~~|_____|~~~~~|_____|~~~~~|_____|~~~~~|_____
---- POD_IN_buf_l         >|<    ..   >|<    D0   >|<    D2   >|<    D4   >|
---- POD_IN_buf_h         >|<    ..   >|<    D1   >|<    D3   >|<    D5   >|

  -- ダブル・データレート・バッファ
  altera_ddr_inst : altera_ddr PORT MAP (
             datain    => (POD2 & POD1),
             inclock   => POD_clk,
             dataout_h => POD_IN_buf_h,
             dataout_l => POD_IN_buf_l);

  -- 1：4デシリアライザ
  deser_inst : DESERx4 PORT MAP (
             DIN_L   => POD_IN_buf_l,
             DIN_H   => POD_IN_buf_h,
             inclock => POD_clk,  -- Max 400MHz
             outclk  => POD_qclk, -- Max 100MHz
             reset   => DPRAMCtrlPODReg(2),
             OUTDATA => POD_OUTD);

    PROCESS (POD_qclk) IS
      BEGIN
        IF (POD_qclk'event AND POD_qclk ='1') THEN
          POD_IN(1) <= POD_OUTD( 15 downto   0);
          POD_IN(2) <= POD_OUTD( 31 downto  16);
          POD_IN(3) <= POD_OUTD( 47 downto  32);
          POD_IN(4) <= POD_OUTD( 63 downto  48);
          POD_IN(5) <= POD_OUTD( 79 downto  64);
          POD_IN(6) <= POD_OUTD( 95 downto  80);
          POD_IN(7) <= POD_OUTD(111 downto  96);
          POD_IN(8) <= POD_OUTD(127 downto 112);
        END IF;
      END PROCESS;

（中略）
```